Inf Retrieval J (2015) 18:95–122
DOI 10.1007/s10791-014-9249-4

Improving daily deals recommendation using explore-then-exploit strategies

Anisio Lacerda · Rodrygo L. T. Santos · Adriano Veloso · Nivio Ziviani

Received: 4 May 2014 / Accepted: 6 December 2014 / Published online: 21 December 2014

Abstract Daily-Deals Sites (DDSs) enable local businesses, such as restaurants and stores, to promote their products and services and to increase their sales by offering customers significantly reduced prices. If a customer finds a relevant deal in the catalog of electronic coupons, she can purchase it and the DDS receives a commission. Thus, offering relevant deals to customers maximizes the profitability of the DDS. An immediate strategy, therefore, would be to apply existing recommendation algorithms to suggest deals that are potentially relevant to specific customers, enabling more appealing, effective and personalized catalogs. However, this strategy may be innocuous because (1) most of the customers are sporadic bargain hunters, and thus past preference data is extremely sparse, (2) deals have a short living period, and thus data is extremely volatile, and (3) customers' taste and interest may undergo temporal drifts. In order to address such a particularly challenging scenario, we propose a new algorithm for daily deals recommendation based on an explore-then-exploit strategy. Basically, we choose a fraction of the customers to gather feedback on the current catalog in the exploration phase, and the remaining customers to receive improved recommendations based on the previously gathered feedback in a posterior exploitation phase. During exploration, a co-purchase network structure is updated with customer feedback (i.e., the purchases of the day), and during exploitation the updated network is used to enrich the recommendation algorithm. An advantage of our approach is that it is agnostic to the underlying recommender algorithm. Using real data

A. Lacerda · R. L. T. Santos (✉) · A. Veloso · N. Ziviani
Department of Computer Science, Universidade Federal de Minas Gerais, Belo Horizonte, Brazil
e-mail: rodrygo@dcc.ufmg.br

A. Lacerda
e-mail: anisio@dcc.ufmg.br

A. Veloso
e-mail: adrianov@dcc.ufmg.br

N. Ziviani
e-mail: nivio@dcc.ufmg.br; nivio@zunnit.com

N. Ziviani
Zunnit Technologies, Belo Horizonte, Brazil

obtained from a large DDS in Brazil, we show that the way in which we split customers into exploration and exploitation impacts by large the effectiveness of the recommendations. We evaluate different splitting strategies based on network centrality metrics and show that our approach offers gains in mean average precision and mean reciprocal rank ranging from 14 to 34 % when applied on top of state-of-the-art recommendation algorithms.

Keywords Daily-deals sites · Recommender systems · Armed bandit setting

1 Introduction

The daily deals model is based on the group-buying concept, which allows local businesses to advertise their products and services while providing customers with significantly discounted offers (aka deals). The typical marketing strategy of online group-buying systems, like Groupon[1] and LivingSocial,[2] is to use different communication channels, mainly emails and short messages, to alert registered customers about products and services at big discounts for a limited time. Every time a customer purchases a deal featured in the alert, the daily-deals site (or simply DDS) earns a commission. Thus, offering appealing and relevant deals to the customers increases the odds of purchases, making an important contribution to maximizing the profitability of the DDS.

Selecting deals that are relevant to specific customers might be viewed as a recommendation problem, for which a variety of algorithms are available (Shi et al. 2012; Rendle et al. 2009; Hu et al. 2008; Pan et al. 2008). Collaborative filtering algorithms, in particular, rely on past preference data in order to provide an effective recommendation performance (Ricci et al. 2010; Koren 2010). The daily-deals recommendation scenario, however, shows characteristics that make the problem particularly challenging. First, most of the registered customers of a DDS are sporadic bargain hunters, and thus past preference data is extremely sparse and noisy. Also, deals have a short living period, and thus data is extremely volatile. Finally, customers' taste and interest may undergo temporal drifts as new deals appear (Byers et al. 2012a; Lappas and Terzi 2012). Previous results demonstrate the effectiveness of recommendation algorithms in application scenarios as diverse as movie (Miller et al. 2003) and music recommendation (Aizenberg et al. 2012; Koenigstein et al. 2011), scientific paper assignment (Conry et al. 2009), and tag recommendation (Menezes et al. 2010), but leave open the question of whether existing recommendation algorithms are effective for the challenging scenario posed by daily-deals recommendation. More specifically, it is still unclear whether these algorithms are able to improve over the naive strategy based on sending non-personalized email messages with ordinary deals.

In this paper, we propose a novel daily-deals recommendation approach which addresses the aforementioned challenges using an explore-then-exploit strategy to improve the recommendations produced by a baseline recommender algorithm. In order to acquire and use customer feedback, we separate customers into two partitions: the exploration and exploitation sets. Customers in the exploration set receive all the deals of the day, which

[1] http://www.groupon.com.

[2] http://www.livingsocial.com.

are displayed to them in random order. Rather than improving deal recommendation, exploration focuses on capturing customer interaction by gathering the most timely feedback (i.e., the purchases of the day). Then, all feedback captured during exploration is leveraged to improve the recommendations for the remaining customers in the exploitation set.

Our approach continuously gathers customers' feedback, augmenting a co-purchase network structure which evolves as deals are purchased. In the co-purchase network, each node represents a customer and an edge between two nodes is created once the corresponding customers purchase the same deal. Intuitively, a deal becomes more likely to be relevant to a particular customer if she is close to others that have also purchased this deal recently. Following this intuition, we modify the output of existing recommendation algorithms, whether based on popularity or collaborative filtering, to weigh more heavily deals that were purchased by nearby customers. As a result, the historic purchase behavior of a target customer can be used to locate customers with similar interests and that have purchased deals in the current catalog.

The explore-then-exploit strategy (Kaelbling et al. 1996) is a way to cope with taste drift and sparse and volatile data since suggestions are performed based on recently gathered feedback. On the other hand, an important aspect that greatly impacts the effectiveness of our approach is the decision of which customers will compose the exploration and exploitation sets. The key factor is to acquire new evidence during the exploration phase and to optimize choices of recommendations based on the existing evidence during the exploitation phase. Therefore, it is necessary to improve recommendations based on the knowledge already acquired with the gathered feedback, while attempting new actions to further increase and update knowledge. Clearly, neither a purely exploration nor a purely exploitation approach works best in general, and a good trade-off is needed.

Our explore-then-exploit approach explores customers that (1) are likely to provide feedback, and (2) are close to the largest number of customers in the co-purchase network. We also exploit the gathered feedback with customers that are likely to benefit from such information. Intuitively, the more connected a customer is the more feedback she may provide (as an exploration customer) and the more feedback she may benefit from (as an exploitation customer). Therefore, we employ network centrality metrics (Boccaletti et al. 2006) to aid the decision of which customers will compose the exploration and exploitation sets. The objective in this case is to separate customers in a way that ensures that enough feedback is captured during the exploration phase and used during the exploitation phase. We show that alternating central customers into exploration and exploitation is better than fully exploring or exploiting central customers.

To evaluate our proposed approach, we conduct a systematic set of experiments using real data obtained from Peixe Urbano,[3,4] the largest DDS in Brazil. The experiments show that our proposed explore-then-exploit approach is effective for daily-deals recommendation, providing mean average precision and mean reciprocal rank (MRR) improvements that range from 14 to 34 % on top of state-of-the-art recommendation algorithms from the literature. Further analysis reveals that an appropriate sorting of the available customers and their separation into exploration and exploitation sets are crucial for the success of our approach.

[3] http://www.peixeurbano.com.br.

[4] Preliminary results on strategies for splitting customers into exploration and exploitation were presented by Lacerda et al. (2013) and are used as a baseline for our investigations in Sect. 4.

The remainder of this paper is organized as follows. In Sect. 2, we review the related literature on DDSs and on exploration-exploitation recommendation strategies. In Sect. 3, we describe our explore-then-exploit approach. In Sect. 4, we detail the dataset, baselines, evaluation methodology and experimental results obtained. In Sect. 5, we further analyze the efficiency of our proposed approach. Finally, in Sect. 6, we present concluding remarks and directions for future work.

2 Related work

In this section, we discuss relevant work on DDSs and on recommendation systems based on the exploration-then-exploitation concept.

2.1 Daily-deals recommendation

The success of DDSs has motivated research on several phenomena in the context of daily-deals such as data-driven analysis and economic models. The focus of data-driven analysis is to identify and understand which are the key characteristics of DDSs from their purchase history. Economic models have been built to mainly understand the business model of DDSs. Byers et al. (2011), for instance, presented evidence that Groupon is behaving strategically to optimize deal offerings, giving customers incentives to make a purchase (e.g., deal scheduling and duration, deal featuring, and limited inventory) other than price. In addition, Byers et al. (2012a) provided an extensive investigation of the daily-deals domain and the social and business aspects associated with it.

Several studies have been conducted about the propagation effect of daily deals on social networks considering, for instance, the impact that a deal has on the merchant's subsequent ratings on social review sites such as Yelp. Different from previous research that observed a decrease in the ratings for merchants using Groupon (Byers et al. 2011), Byers et al. (2012b) argued that this effect was overestimated. Kumar and Rajan (2012) studied, among other economic aspects, the profitability of social coupons and concluded that they yield profits for local businesses.

Considering economic aspects of DDSs, Arabshahi (2011) studied the business model of Groupon. An empirical analysis of the experience of businesses that used Groupon was conducted by Dholakia (2010). Models for evaluating the benefits and the drawbacks of Groupon when considering the merchants' point of view were proposed by Edelman et al. (2011). Kauffman and Wang (2001) conducted an extensive study of the underlying dynamics of the group-buying discount market. Liu et al. (2013) investigated how website cues affect personality traits that trigger online impulse purchases. In particular, they argued that customers are more impulsive when participating in the buying event in daily-deals sites because a deal is typically available only for a certain period of time and may not be available again in the future. To model customers' personality traits, they proposed a behavioral model derived from prior discussion on online and in-store impulse buying. From an evaluation questionnaire, they concluded that personality factors, for instance, gratification, normative evaluation, and impulsiveness are key determinants to trigger impulse purchases, while perceived website cues of visual appeal, website ease of use, and product availability are important precursors. Different from the aforementioned works that focused on understanding the DDS business, we investigate how to improve daily-deals recommendation effectiveness.

There are two algorithmic problems in the DDS recommendation scenario: (1) deal-size estimation and (2) deal ordering for revenue maximization. The deal size estimation refers to the number of coupons that a DDS is expecting to sell for a given deal. Byers et al. (2012a) modeled the deal-size estimation as a linear combination of features associated with deals and used ordinary least squares regression to fit their model. Another approach to this estimation problem was given by Ye et al. (2012): instead of using a linear regression of deal attributes, they modeled the popularity of deals as a function of time.

The deal ordering for revenue maximization is about daily-deal selection and scheduling: given a set of candidate deals, select the ones that should be featured in order to maximize the DDS revenue. Lappas and Terzi (2012) analyzed their daily-deal datasets and provided two different formulations for this problem: (1) deal selection, which handles the selection of a set of deals for a single day, and (2) deal scheduling, which stands for a long-term scheduling strategy for multiple days. Notice that in both formulations it is assumed that the deal size can be accurately estimated. In contrast to these approaches, which attempt to identify a catalog of recommendable deals, we tackle the problem of how to provide a personalized recommendation of deals from a given catalog.

2.2 Exploration-exploitation recommendation strategies

The decision of which customers will compose the exploration and exploitation sets, known as the exploration-exploitation dilemma, has been an important subject in the Reinforcement Learning literature (Sutton and Barto 1998). This dilemma refers to a decision-maker that needs to sequentially choose between learning about the current options (exploration) or following what seems to be the best option at the moment (exploitation). Since the current best option may be sub-optimal, the decision-maker explores hoping to discover an option that beats the current best option.

Robbins (1952) presented the Multi-Armed Bandits (MAB) setting (aka K-armed bandits) as a method to deal with the exploration-exploitation dilemma. Historically, the name "bandit" comes from an imaginary gambler playing a K-slot machine in a casino. The gambler may pull an arm from any of the slot machines. Each time an arm is pulled, a random reward, independent from any previous rewards, is returned. The arms are also assumed to be independent from one another. The gambler's objective is to accumulate as much reward as possible in the long run. Formally, the MAB setting is described as follows. Let K refer to the number of arms that the decision-maker can pull. At each time step t, the decision-maker pulls arm i_t and an associated reward r_{i_t} is returned. The rewards are drawn from an unknown distribution of arm i_t. The process repeats for a horizon $T > 0$ until it finishes. The decision-maker's objective is to maximize the sum of rewards. A MAB algorithm is a strategy that determines the sequence of arms chosen in order to achieve a maximal reward at each time step t and, ultimately, maximize the cumulative reward (or, equivalently, to minimize the cumulative regret of bad decisions).

There are two lines of work devoted to study MAB algorithms: (1) context-free and (2) context-aware algorithms. The context-free MAB problem has been investigated by several researchers (Robbins 1952; Lai and Yakowitz 1995; Auer et al. 2002; Even-Dar et al. 2006; Radlinski et al. 2008). In this line of work, MAB algorithms take into account only information about the arms when making decisions, and greedy strategies are often used to select arms. Watkins (1989) presented the ϵ-greedy algorithm which employs the ϵ parameter in order to balance exploration and exploitation. In particular, the decision-maker follows the greedy algorithm (i.e., she chooses the arm with the

highest reward estimate) with probability $(1-\epsilon)$, and a random arm with probability ϵ. Auer et al. (2002) presented a class of MAB algorithms called *upper confidence bound* that have optimal asymptotic convergence. A drawback of these algorithms is that exploration becomes difficult as the space of possible solutions is much larger (Mannor and Tsitsiklis 2004).

Azoulay-Schwartz et al. (2004) investigated the task of buying a particular product by an automated agent. They assume that there are several potential suppliers of this product, which differ from one another in terms of quality and price charged. The authors showed that this problem can be modeled as a special case of the MAB problem. Their decision making problem refers to which supplier should be selected. In contrast, in our work, we are interested in selecting relevant deals to customers to maximize the profitability of a DDS.

Another strategy for context-free MAB algorithms is the budget-limited one, first presented by Guha and Munagala (2007) and then generalized by Tran-Thanh et al. (2010). This strategy assumes that actions are costly, and constrained by a fixed budget. As the budget is predefined, the duration of the problem is finite, and the ideal exploitation is not obtained by pulling the optimal arm repeatedly, but combinations of arms that maximize the reward within the budget. Consequently, the reward for all arms must be estimated, as any of them can appear in the optimal combination. A common strategy within this scenario is the ϵ-first strategy, where ϵ of the budget is used to learn the arms' rewards (exploration) and $1-\epsilon$ is used for exploitation. In this approach, exploration and exploitation are performed in separate phases, that is, only when exploration finishes the exploitation phase starts. In this paper, we propose an ϵ-first approach, which is described in details in Sect. 3.

Context-aware MAB algorithms (aka contextual bandits, associative bandits, bandits with side information, bandits with co-variate, and associative reinforcement learning) were recently proposed and successfully applied in a variety of areas, such as Web content optimization (Li et al. 2010a), online advertisement (Chakrabarti et al. 2008; Li et al. 2010b), and recommender systems (Bouneffouf et al. 2012). Most of the algorithms applied to recommendation follow this context-aware approach, and model the recommeSDFndation problem using a MAB setting in which the items to be recommended are the arms and the users, represented as feature vectors, form the context. In particular, the user (context) is revealed at each time step. For instance, Bouneffouf et al. (2012) considered contextual changes in the geographical, temporal, and social dimensions to improve recommendations in a mobile setting. Other interesting works include the one by Li et al. (2010a), who presented a MAB algorithm that incorporates context information to suggest news articles to users.

Chakrabarti et al. (2008) described a contextual advertising system that combines ad relevance with click information using a logistic regression model. Li et al. (2010b) presented another work in contextual advertising systems focusing on strategies to balance the long-term impact of exploration and exploitation on the system performance. Mahajan et al. (2012), in turn, proposed an *upper confidence bound*-based contextual bandit algorithm to estimate the average rating for comments. Relatedly, He et al. (2013) investigated online learning strategies for auction mechanisms for sponsored search. They showed that this task corresponds to a new type of multi-armed bandit problem, in which dependent arms share information gathered during the exploration phase.

Our proposed approach greatly differs from all aforementioned works in several ways. First, given that the daily-deals recommendation scenario comprises many more customers

(or users) than valid deals (or items), we model customers as arms, instead of the typical strategy of modeling items as arms. Second, instead of following an explicit context-aware strategy, we employ an alternative solution based on a co-purchase network structure and rely on network centrality metrics in order to find a proper balance between exploration and exploitation. Third, the recommendation scenario we consider is extremely dynamic in the sense that the deals of the day span a very short lifetime (typically, a deal stays in the catalog for four days), since deals appear frequently but also disappear frequently, and this needs also to be considered.

3 Explore-then-exploit recommendation

As discussed in the previous sections, each day, a DDS alerts its customers about current deals that may be of interest to them. In this section, we introduce our explore-then-exploit bandit recommendation approach aimed to improve the effectiveness of such alerts. Our bandit setting is presented next:

- Arms: we want to determine the sequence of customers that will incur the largest number of purchases.
- Reward: at time-step t, deals are presented to customer c_t. The reward function is defined as follows:

$$r_{c_t} = \begin{cases} 1, & \text{if customer } c_t \text{ purchased a deal} \\ 0, & \text{otherwise} \end{cases}$$

- Context: a co-purchase network is used as context in order to sort customers. The network is updated after each purchase is performed.
- Horizon: the horizon is fixed, in the sense that the same universe of customers are considered daily.

In particular, each day, our proposed approach performs the following sequential steps:

Sorting Customers We start the process by sorting customers that (1) are more likely to provide feedback, and (2) share similar tastes with many others (i.e., their feedback is likely to benefit other cutomers).

Splitting Strategy Once we sort customers, we are interested in splitting them into two distinct sets: (1) customers from whom we will obtain feedback on the current catalog, and (2) customers that will receive improved recommendations based on the previously gathered feedback.

Exploration Once we determine the set of customers that will provide feedback on the current catalog, we send all deals of the day to these customers, and store their click feedback. The deals are displayed to the customers in random order.[5]

Exploitation Once we gather feedback on the current catalog, we use this feedback to send improved recommendations to exploitation customers.

In the following sections, we detail each step of our approach.

[5] More specifically, we randomized the order in which deals are displayed to the customers, and a purchase is only considered if the purchased deal appears within the items that were originally seen by the customer.

3.1 Sorting customers

This step starts by building a co-purchase network as illustrated by a small example in Fig. 1. The co-purchase network is built from historical purchase data. The network is represented by an unweighted undirected graph, where each node represents a customer and each edge connecting two customers represents a deal purchased by both customers. Note that an edge is created when at least one deal was purchased by both customers.

In Fig. 1, we show six purchase records involving six customers and three deals. For instance, customer c_{37} bought deals d_2 and d_{20}. This purchase history data leads to the co-purchase network shown in Fig. 1. As an example, customers c_{10} and c_{19} bought deal d_1, thus they are connected in the co-purchase network. The connected customers are then sorted according to their degree in the co-purchase network.

The criterion used to sort customers in the co-purchase network must satisfy two desired properties, i.e., we want to separate customers that (1) are more likely to provide feedback (i.e., to purchase a deal), and (2) share similar tastes with many other customers (i.e., their feedback is likely to benefit other customers).

These two properties are more evident in customers that are central in the co-purchase network, where central means customers that have a higher score according to a given network centrality metric as described in the following. In our experiments in Sect. 4, we employ five representative network centrality metrics (Freeman 1979; Bonacich 1987; Borgatti 2005) to sort customers:

- The *Degree* of a customer is defined as the number of customers she is directly connected to in the co-purchase network, i.e., the number of other customers that have a taste similar to hers. This metric may also convey the prior probability of purchase of a customer (the higher the degree, the higher the odds of a purchase).
- The *Betweenness* of a customer, which is also called socio-centric betweenness centrality, is computed as the fraction of shortest paths between all pairs of customers that pass through the customer of interest. As expected, customers that have a high probability of occurring on a randomly chosen shortest path between any two customers are said to have high betweenness centrality. A formal definition of the betweenness centrality b_c of target customer c is given by the summation of the number of geodesic paths between any two nodes s and t through c, normalized by the total number of geodesic paths between s and t, given by:

$$b_c = \sum_{s \neq c \neq t} \frac{\theta_c^{st}}{\theta^{st}}, \tag{1}$$

  where θ^{st} is the total number of shortest paths from node s to node t and θ_c^{st} is the number of those paths that pass through c.
- The *Eigenvector centrality* of a customer is defined in a circular manner. In particular, the eigenvector centrality of a node is proportional to the sum of the centrality values of all its neighbors. In our co-purchase network, a central node is characterized by her connectivity to other important nodes. In this case, a central customer corresponds to a well-connected node and has a dominant influence on the surrounding sub-graph. We specify the eigenvector centrality in terms of the adjacency matrix of our co-purchase network. Let v_i be the i-th element of vector v, representing the centrality of node i, where N_i is the set of neighbors of node i and let A be the $n \times n$ adjacency matrix of the undirected co-purchase network. In matrix notation we have:

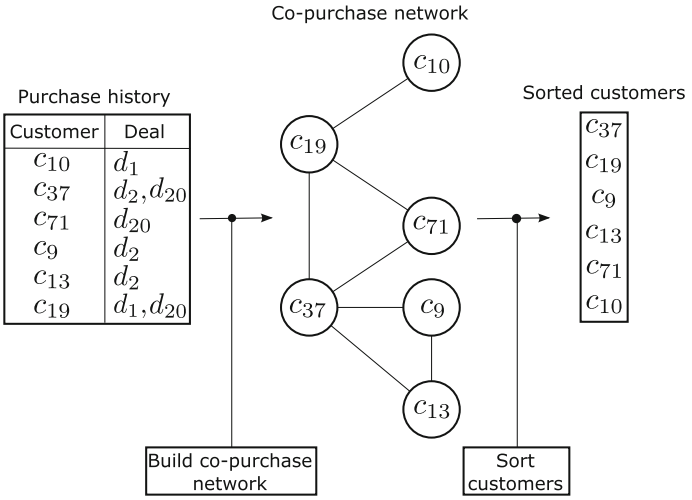


Fig. 1 Sorting customers according to their purchase history data

$$Ae = \lambda e, \tag{2}$$

where e is an eigenvector of A, and λ is its associated eigenvalue. Thus, the eigenvector centrality x_i of node i is defined as the i-th entry in the normalized eigenvector belonging to the largest eigenvalue of A.

– The *PageRank* algorithm was originally proposed to rank Web documents by analyzing the structure of hyperlinks between them (Page et al. 1999). In our context, the *PageRank* of a customer is defined recursively and depends on the value of the *PageRank* of her neighbors. Formally, the *PageRank* of a customer c is given by:

$$PR_c = \alpha \sum_{c_j \in N_c} \frac{PR_{c_j}}{|N_{c_j}|} + \beta, \tag{3}$$

where α is the probability that some arbitrary customer (a node in the co-purchase network) is chosen, β is the probability that a random customer is chosen, N_c denotes the neighborhood of customer c in the co-purchase network and, for a given neighbor $c_j \in N_c$, $|N_{c_j}|$ denotes the degree of c_j. Normalizing by degree ensures that we obtain a stochastic matrix (either all the columns or all the rows sum to one).

– The *Clustering Coefficient* of a customer denotes how close her neighbors are to forming a clique (i.e., a complete graph). Formally, it is defined as the ratio between the total number of connections among the neighbors of target customer c and the total number of possible connections between her neighbors, which is given by:

$$C_c = \frac{L_c}{\binom{n}{2}}, \tag{4}$$

where L_c is the number of actual links between the neighbors of c, and n stands for the number of neighbors of c.

Centrality measures may present differing results with respect to the suggested importance of customers. A comparison involving the most used centrality measures can be found in

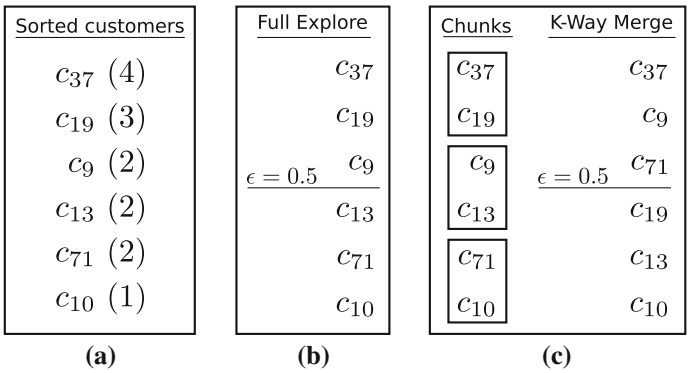


Fig. 2 Strategies for balancing the exploration and exploitation sets. Fully exploring central customers, or merging them into exploration and exploitation. In this example, we use $k = 2$ (i.e., the size of each chunk) in the k-Way Merge strategy

Landherr et al. (2010). Further, centrality measures are also related to sampling approaches for estimating properties of graphs (Leskovec and Faloutsos 2006; Maiya and Berger-Wolf 2011).

3.2 Splitting customers

Having a sorted list of customers obtained in the previous step, we separate them into exploration and exploitation sets. To this end, we propose two splitting strategies, called Full Explore (FE) and k-Way Merge. In Fig. 2, we illustrate the two proposed splitting strategies following the same previously presented small example. The input to both strategies is shown in Fig. 2a and refers to the array of customers sorted by their degree in the co-purchase network, produced as the output of the sorting step (see Fig. 1).

The FE strategy separates customers following the order imposed by the corresponding centrality metric. In this case, most central customers (wrt. a specific centrality metric) receive email messages first, and are thus fully explored. This is the strategy initially proposed by Lacerda et al. (2013), and is used as a baseline in this work. The FE splitting strategy is inspired by the ϵ-first MAB algorithm, where a pure exploration phase is followed by a pure exploitation phase. In this case, given a total of N customers, the exploration phase comprises $\epsilon \times N$ customers, while the exploitation phase includes the remaining $(1 - \epsilon) \times N$ customers. In Fig. 2b, we present the final split returned by the FE strategy. In particular, considering $\epsilon = 0.5$, we equally split customers between the exploration and exploitation sets. Hence, the exploration set is given by customers c_{37}, c_{19}, and c_9 and the exploitation set is given by customers c_{13}, c_{71}, and c_{10}.

The FE strategy considers the customers sorted by the original sorting criterion (e.g., Degree) and splits them according to the given ϵ value. However, the strategy of fully exploring central customers has a limitation. As the co-purchase network is expected to have a very low diameter, central customers have a high probability of being already connected. For instance, if we explore the feedback given by the two most central customers in the network, the probability of both having a high number of common neighbors is very high. In this case, exploring both customers might be ineffective since their feedback is likely to be redundant. In fact, feedback from only one of these customers

would be enough, as it benefits a set of customers very similar to the one influenced by the feedback of the other customer. In addition, central customers themselves are likely to benefit the most during exploitation.

To avoid the possible redundant feedback introduced by the FE splitting strategy, we propose to alternate central customers into exploration and exploitation sets using a k-Way Merge (KWM) strategy. Specifically, customers are arranged into chunks, so that some chunks are composed by central customers while other chunks contain more peripheral customers. These chunks are then merged into a single list of customers by alternating through them in a round-robin fashion, ensuring that both exploration and exploitation sets contain central customers. The KWM strategy also uses the ϵ-first approach to split customers between the exploration and exploitation sets, as illustrated in Fig. 2c. In the figure, we present the two phases of the KWM strategy. First we consider that the size of each chunk is 2, i.e., $k = 2$. Hence, we have three chunks, namely, (1) c_{37}, c_{19}, (2) c_9, c_{13}, and (3) c_{71}, c_{10}.[6] On the right side of Fig. 2c, we present the resulting merge of these chunks. Here, we also consider $\epsilon = 0.5$ and split the final merged rank half for exploration and half for exploitation. Hence, the KWM strategy returns customers c_{37}, c_9, and c_{71} for exploration and customers c_{19}, c_{13}, and c_{10} for exploitation. In this example, we avoid a possible redundant feedback from customers c_{37} and c_{19} because they are now in different sets.

3.3 Exploration

Exploration data is used to model customer behavior with regard to the deals appearing in the current catalog. More specifically, exploration data is used to answer counterfactual questions (Bottou et al. 2013) regarding how deals should be displayed to the customers in order to improve daily-deals recommendation performance. In order to mitigate bias, exploration data is gathered as follows:

1. a set of customers are separated and will be explored. Thus, these customers receive all deals of the day displayed in random order,
2. a purchase is only considered as feedback if the purchased deal is within the items that were originally seen by the customer. That is, we have recorded the number of deals examined by each customer each day, and the feedback is only considered if the purchased deal is within the items examined by the customer.

The Exploration phase consists of obtaining feedback from customers with regard to the deals of the day. We sampled purchase data in order to simulate a scenario in which customers in the exploration set receive a set of deals displayed in random order. After that, the algorithm waits for feedback from these customers. Having gathered feedback from customers in the exploration set, the co-purchase network is updated and the exploitation phase starts. In Fig. 3, we illustrate the exploration phase of our approach. Here we consider the FE splitting strategy to select customers for the exploration set. The customers selected for exploration are c_{37}, c_{19}, and c_9. For each time step l, where $1 \leq l \leq 3$, we select the target customer c_l to send the current catalog and capture her feedback f_l. The feedback is used to update the co-purchase network. For instance, in Fig. 3, we create two edges between customers (c_{19}, c_{71}) and (c_9, c_{10}).

[6] Note that if the number of customers is not divisible by k, the last chunk will contain less than k customers.

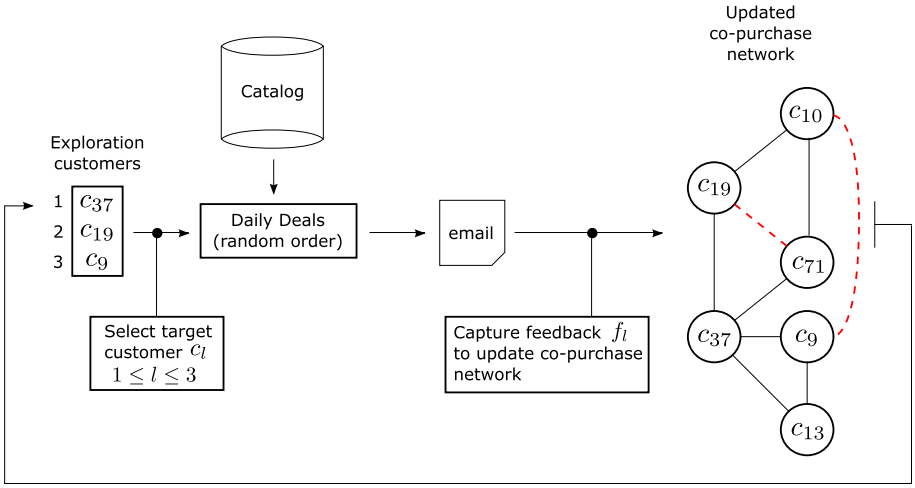


Fig. 3 Customers in the exploration set receive the deals of the day displayed in random order (by simulation). The entire process is repeated for each target customer c_l, where $1 \leq l \leq 3$. Note that, since our approach follows the ϵ-first approach, the acquired feedback is not leveraged until the end of the exploration phase

3.4 Exploitation

Having sent recommendations to all customers in the exploration set, and having received their feedback, we start the Exploitation phase of our approach. In this phase, all exploitation customers receive a message featuring a list of deals. In particular, we model the deal recommendation during the exploitation phase as a top-N recommendation task (Cremonesi and Koren 2010), and consider $N = 5$, i.e., we send 5 deals per email. Given the ranked list of deals initially produced by the baseline recommendation algorithm for a particular customer, we incorporate the feedback obtained in the co-purchase network in order to further personalize the ranking toward the potential interests of this customer. In Fig. 4, we illustrate the exploitation phase of our approach. Here we also consider the FE splitting strategy to select customers for the exploitation set. The customers selected for exploitation are c_{13}, c_{71}, and c_{10}. For each time step l, where $4 \leq l \leq 6$, we select the target customer c_l to send recommendations.

Different from the process of exploration, in the exploitation phase, we use the updated co-purchase network to re-rank the recommendations built by the recommender algorithm. Precisely, given a target customer c and the set of deals D currently available in the catalog, we produce a re-ranking $\pi(c, D)$ by sorting the deals $d \in D$ by their decreasing neighborhood score $s_n(c, d)$ and baseline score $s_b(c, d)$, the latter serving as a tie-breaking criterion, such that:

$$\pi(c, D) = sort_{\langle s_n(c,d), s_b(c,d)\rangle}\{d \in D\}, \tag{5}$$

where $s_b(c, d)$ is the score assigned to deal d with respect to customer c by the baseline recommender, whereas $s_n(c, d)$ is the neighborhood score assigned to the pair $\langle c, d\rangle$, according to:

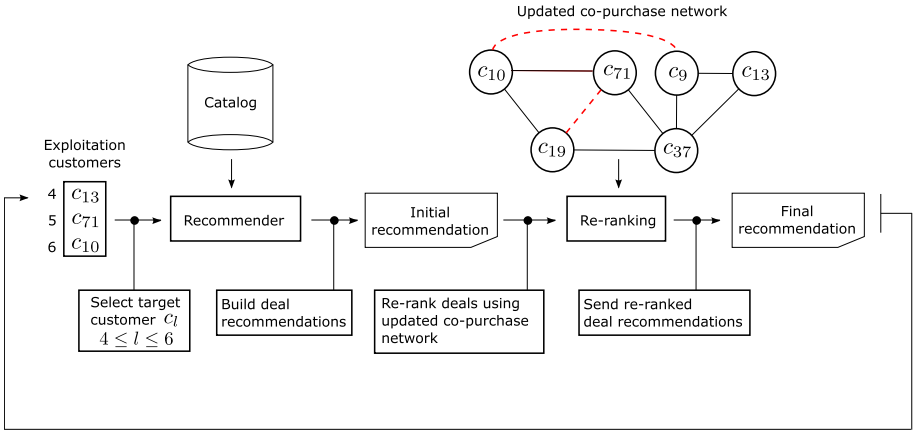


Fig. 4 Recommendation process for customers in the exploitation set. The entire process is repeated for each target customer c_l, where $4 \leq l \leq 6$

$$s_n(c,d) = \frac{\sum_{c_j \in N_c} \mathbb{1}(c_j, d)}{|N_c|}, \tag{6}$$

where N_c denotes the set of neighbors of customer c in the updated co-purchase network, and $\mathbb{1}(c_j, d)$ is an indicator function, denoting whether customer $c_j \in N_c$ has purchased deal d.

In order to illustrate the re-ranking procedure, we present a toy example. We assume that most popular (MP) is the recommender algorithm and its ranking is composed by 5 deals sorted as $D = \{d_2, d_{20}, d_5, d_7, d_1\}$. We also assume that (1) the *exploration set* is given by customers $\{c_1, c_7\}$ and (2) the *exploitation set* is given by customer $\{c_3\}$. The neighborhood of target customer c_3 in the updated co-purchase network is given by customers $\{c_1, c_7, c_8\}$. Since customer c_8 is not present in the *exploration set*, she will not be considered during re-ranking. Finally, let us assume that (1) customer c_1 provides feedback on deals $\{d_5, d_7\}$ and (2) customer c_7 provides feedback on deal $\{d_7\}$. Thus, we are able to build the new ranking $D' = \{d_7, d_5, d_2, d_{20}, d_1\}$. We used the feedback provided by customers c_1 and c_7 to update the rank of deals d_7 and d_5 to top positions. Note that the relative order of the deals $\{d_2, d_{20}, d_1\}$ ranked by the MP recommender algorithm was preserved.

Exploration and exploitation are sequentially performed, i.e., every testing day starts with the exploration phase, which is then followed by the exploitation phase. Specifically, we consider two criteria to decide when to start the exploitation phase: (1) if the time of day reaches 12:00 pm or (2) if the percentage of feedback returned is greater than 98 % of the feedback asked from customers. We choose the criterion that is satisfied first. In the end of each testing day, all purchases are used to update the historic data H, and the explore-then-exploit process is repeated. By relying on collaborative purchase information, exploitation may suffer from the cold-start problem that is typical to collaborative filtering approaches (Schein et al. 2002). This is particularly the case when producing recommendations for new users, which by large outnumber the available items in a DDS setting on any given day. While probing part of the user base for further feedback during the exploration phase helps mitigate the problem during exploitation, in an extreme situation, a

user with no past purchase information will only receive non-personalized recommendations. In the next section, we assess the recommendation effectiveness of our approach compared to state-of-the-art collaborative filtering recommenders on a large sample of usage data from a commercial DDS.

4 Experimental evaluation

In this section, we assess the effectiveness of our proposed approach. In Sect. 4.1, we describe the evaluation methodology, the recommendation baselines, and the DDS dataset used in our investigations. The analysis of our experimental results follows in Sect. 4.2, with a discussion of the recommendation performance of our approach and the impact of the several sorting criteria and splitting strategies introduced in Sect. 3. An analysis of the time complexity of all the components in our proposed approach is later presented in Sect. 5.

4.1 Experimental setup

4.1.1 Evaluation methodology

Our evaluation follows the Interleaved Test-Then-Train methodology (Bifet et al. 2010). We first evaluate all recommendations a customer receives on day t considering as training data all history from the first day until day $t-1$. When considering testing day $t+1$, we follow the same methodology, i.e., we evaluate all recommendations a customer receives on day $t+1$ and we consider as training data all history from first day until day t. This process follows for all testing days.

An effective ranking of deals should place relevant deals in the top positions, since these are the positions more likely to be clicked (Feng et al. 2007). Therefore, we assess the effectiveness of our proposed approach in terms of standard retrieval evaluation metrics, namely, mean average precision (MAP) and MRR (Baeza-Yates and Ribeiro-Neto 2008).

4.1.2 Recommendation baselines

We consider three representative recommendation algorithms as baselines in our investigations, namely MostPopular (or simply MP), WRMF (Hu et al. 2008), and CLiMF (Shi et al. 2012). The MP algorithm is the simplest one, as it sorts deals based on their popularity, that is, more purchased deals are placed first in the ranked list. Thus, deals are sorted in the same way regardless of the target customer.

The Weighted Regularized Matrix Factorization (WRMF) algorithm is the state-of-the-art algorithm for scenarios where there is only implicit feedback. Thus, this is a strong baseline for the DDS scenario, in which we do not have any level of preference for deals from customers. In other words, the customers provide only implicit feedback in the form of clicks or purchases. The WRMF algorithm is a matrix factorization approach that extracts latent factors from implicit feedback. The method is based on traditional Singular Value Decomposition and extends it by introducing regularization to prevent overfitting and weights in the error function to increase the impact of positive feedback. In our experiments, we used the WRMF implementation provided in MyMediaLite v3.0.[7]

[7] http://www.mymedialite.net.

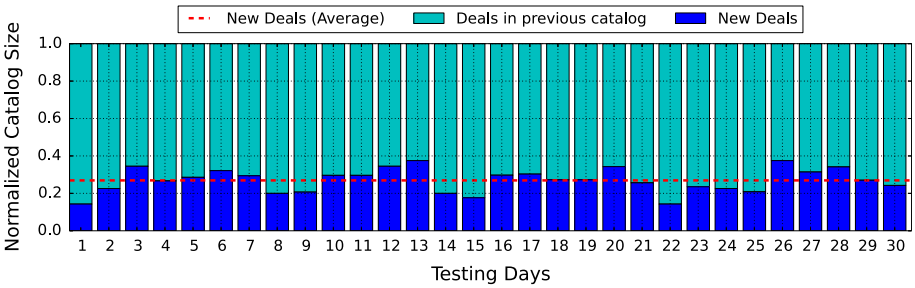
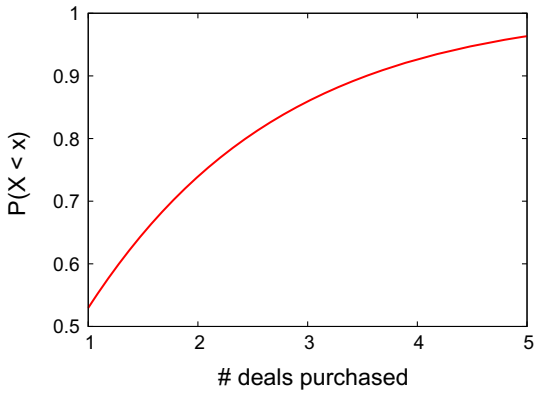


Fig. 5 Dynamic nature of the catalog: number of novel deals for each testing day

Fig. 6 Cumulative distribution of deals purchased in testing days



The third baseline recommender algorithm used was Collaborative Less-is-More Filtering (CLiMF). The algorithm aims to learn latent factors of customers and items by minimizing MRR. CLiMF was proposed to improve the performance of top-N item recommendations in scenarios with the prevalence of binary relevance data and when there are only very few relevant items for any given user. The latter property is particularly pertinent in the DDS domain, where customers rarely buy more than one deal at once. The CLiMF implementation used in our experiments is available from GraphChi v0.2.[8]

4.1.3 Daily-deals dataset

Our experiments were performed using a real-world dataset obtained from Peixe Urbano, the largest DDS in Brazil. The dataset comprises information about deals, customers and purchases, sampled over a period of 2 months in late 2012. Specifically, in this sample, 31,642 customers purchased at least one out of 455 deals, resulting in a total number of 43,274 purchases. The first 31 days were used to build the initial co-purchase network, and the last 30 days were used to test our approach. Note that we build the co-purchase network without using a recommender system when sending deals to customers. Figure 5 illustrates

[8] http://graphlab.org/projects/graphchi.html.

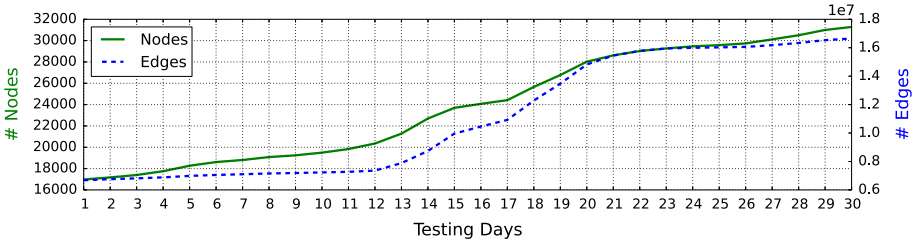


Fig. 7 Co-purchase network evolution: nodes and edges added during the testing days

why DDS scenarios present a big challenge to current recommendation algorithms. Every day, about 28 % of the catalog is composed by new deals, which remain valid for 4 days on average. The pattern during traning days is very similar.

Figure 6 presents the cumulative distribution function of purchases made by customers. From the figure, we can see that more than 50 % of the customers purchased only one deal and approximately 80 % of them purchased at most 2 deals. On average, each customer purchased 1.36 deals, making this dataset extremely sparse if compared with more traditional recommendation datasets, such as Netflix[9] (with 208 ratings per user) and MovieLens 1M[10] (with 166 ratings per user).

Figure 7 shows the evolution of the co-purchase network, in terms of the number of nodes and edges composing the network over time. On the left side, we present the number of nodes (i.e., customers) added, and on the right side, we present the number of edges added (i.e., purchases). We present the evolution of these graph properties as a function of testing days. From the figure, we can see that the number of nodes increases from 16,976 to 31,642 and the number of edges increases from 6,681,262 in the first testing day to 16,648,608 in the last testing day. In the DDS scenario, we have both new customers and new deals available on a daily-basis, and both the number of nodes and the number of edges are always increasing throughout testing days. We can also see that the increment in the number of nodes and edges follows a similar growth rate.

4.2 Experimental results

In this section, we assess the effectiveness of our recommendation approach proposed in Sect. 3. In particular, we aim to answer the following research questions:

Q1. How do existing recommendation algorithms perform in a DDS scenario?
Q2. Can we improve the performance of existing recommendation algorithms using our explore-then-exploit approach?
Q3. What is the impact of different criteria for sorting customers?
Q4. What is the impact of our proposed KWM splitting strategy?

In the remainder of this section, Sect. 4.2.1 addresses questions Q1 and Q2. Questions Q3 and Q4 are addressed in Sects. 4.2.2 and 4.2.3, respectively.

[9] http://www.netflix.com.

[10] http://grouplens.org/datasets/movielens/.

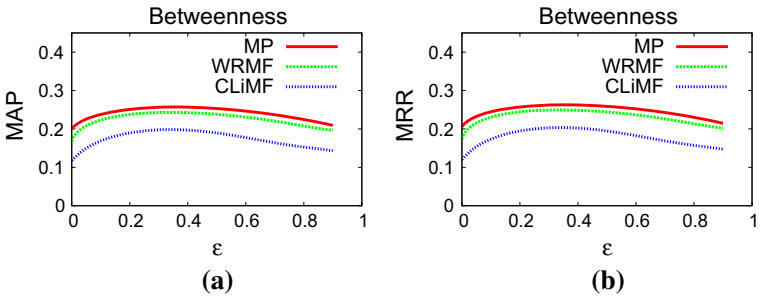


Fig. 8 MAP and MRR results for MP, WRMF, and CLiMF. These results refer to the KWM strategy

4.2.1 Recommendation performance

In order to answer questions Q1 and Q2, Fig. 8 shows the performance of our explore-then-exploit approach on top of three representative recommendation algorithms from the literature, namely, MP, WRMF, and CLiMF. Performance is given in terms of MAP (Fig. 8a) and MRR (Fig. 8b) as we vary the ϵ parameter. Recall that the larger the value of ϵ the more customers are used for exploration. In Fig. 8, our explore-then-exploit approach is applied with the Betweenness sorting criterion [see Eq. (1)] and the KWM splitting strategy, proposed in Sect. 3.2. Results for other sorting criteria and splitting strategies are discussed later in this section.

By contrasting the three recommendation algorithms when $\epsilon = 0$ (i.e., without using our exploration-then-exploitation approach) in Fig. 8a, b, we observe that MP is consistently the most effective, followed by WRMF and then by CLiMF. Recalling question Q1, this result is strikingly different from the performance of these algorithms as reported in the literature. Indeed, WRMF is regarded as the state-of-the-art in traditional recommendation scenarios with implicit feedback (Hu et al. 2008). This may happen because (1) DDS data is extremely sparse, compromising collaborative filtering algorithms, and (2) the daily-deals recommendation scenario is more dynamic than typical recommendation scenarios, in the sense that deals are volatile and new deals are constantly appearing.

Regarding question Q2, we observe that all three recommendation algorithms are improved by our proposed approach as ϵ increases, with a peak around $\epsilon = 0.3$. This result is consistent for different instantiations of our approach. In particular, Fig. 9 provides a breakdown analysis of the performance of our approach in terms of MAP for different sorting criteria (besides Betweenness, we consider Degree, Eigenvector, Clustering Coefficient, and PageRank as sorting criteria, as introduced in Sect. 3.1) in each row and the three considered recommendation algorithms in each column.[11] In addition, besides the overall performance of our approach (i.e., the performance computed for all customers, shown in blue in Fig. 9), we also show its performance on the subset of exploration customers (red in the figure) and on the subset of exploitation customers (green in the figure).

From Fig. 9, we first observe that the performance of the KWM approach on the exploitation customers is naturally superior than its performance on the exploration

[11] Results in terms of MRR show similar trends and are omitted for brevity. A complete analysis in terms of both MAP and MRR is provided in Sects. 4.2.2 and 4.2.3.

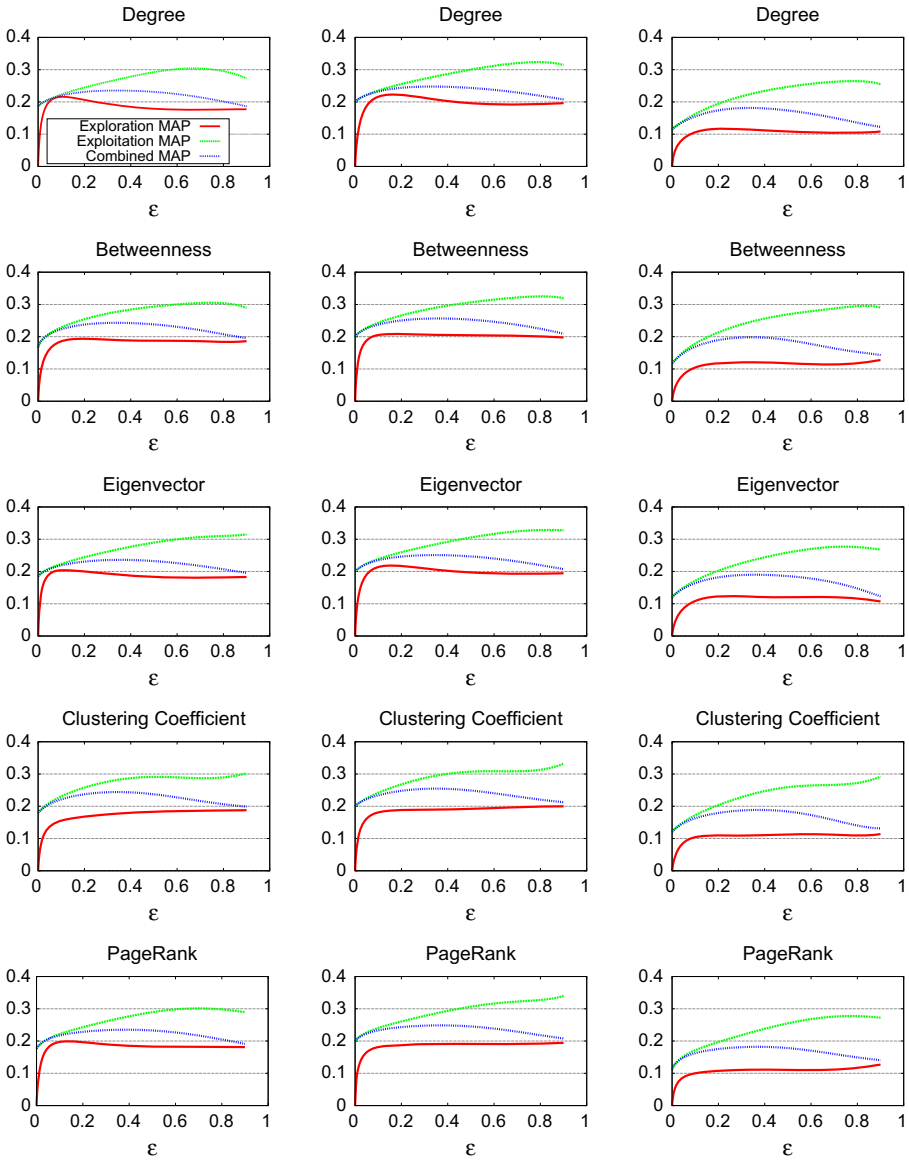


Fig. 9 Exploration, exploitation, and overall MAP results for different configurations. These results refer to the KWM strategy

customers, in which case the underlying baseline recommendation algorithm is applied without any feedback. Indeed, as observed from the red curves, the exploration performance is generally independent of the ϵ parameter. The exploitation performance, on the other hand, increases as the amount of feedback (i.e., the value of ϵ) increases, as depicted by the green curves. This behavior is consistent for all considered sorting criteria and

Table 1 MAP numbers for the MP algorithm

	Random	Degree		Betweenness		Eigenvector		Cluster coefficient		PageRank	
ϵ	–	FE	KWM	FE	KWM	FE	KWM	FE	KWM	FE	KWM
0.00	0.199 •	0.199 •	0.199 •	0.199 •	0.199 •	0.199 •	0.199 •	0.199 •	0.199 •	0.199 •	0.199 •
0.01	○ 0.201 •	○ 0.201 •	△ 0.211 ↑	△ 0.214 ↑	△ 0.213 ↑	○ 0.203 •	△ 0.208 ↑	△ 0.218 ↑	△ 0.209 ↑	○ 0.203 ↑	△ 0.212 ↑
0.05	○ 0.201 •	○ 0.202 •	△ 0.225 ↑	△ 0.224 ↑	△ 0.235 ↑	○ 0.204 •	△ 0.234 ↑	△ 0.242 ↑	△ 0.229 ↑	○ 0.204 ↑	△ 0.232↑
0.10	○ 0.202 •	○ 0.206 •	△ 0.241 ↑	△ 0.240 ↑	△ 0.245 ↑	○ 0.206 •	△ 0.245 ↑	△ 0.249 ↑	△ 0.241 ↑	△ 0.213 ↑	△ 0.242 ↑
0.20	○ 0.201 •	△ 0.213 ↑	△ **0.258** ↑	△ 0.258 ↑	△ 0.259 ↑	△ 0.216 ↑	△ 0.251 ↑	△ 0.252 ↑	△ 0.255 ↑	△ 0.222 ↑	△ 0.247 ↑
0.30	○ 0.200 •	△ 0.224 ↑	△ 0.250 ↓	△ 0.264 ↑	△ **0.267** ↑	△ 0.223 ↑	△ **0.258** ↑	△ 0.248 ↓	△ **0.265** ↑	△ 0.237 ↑	△ 0.251 ↑
0.40	○ 0.201 •	△ 0.231 ↑	△ 0.249 ↓	△ 0.255 ↓	△ 0.259 ↓	△ 0.229 ↑	△ 0.254 ↓	△ 0.248 •	△ 0.263 ↓	△ 0.237 •	△ **0.256** ↑
0.50	○ 0.202 •	△ 0.228 ↓	△ 0.247 ↓	△ 0.252 ↓	△ 0.259 •	△ 0.229 •	△ 0.250 ↓	△ 0.243 ↓	△ 0.254 ↓	△ 0.240 ↑	△ 0.251 ↓
0.60	○ 0.201 •	△ 0.227 ↓	△ 0.241 ↓	△ 0.244 ↓	△ 0.248 ↓	△ 0.226 ↓	△ 0.245 ↓	△ 0.240 ↓	△ 0.240 ↓	△ 0.235 ↓	△ 0.242 ↓
0.70	○ 0.201 •	△ 0.224 ↓	△ 0.230 ↓	△ 0.233 ↓	△ 0.240 ↓	△ 0.220 ↓	△ 0.232 ↓	△ 0.231 ↓	△ 0.229 ↓	△ 0.232 ↓	△ 0.234 ↓
0.80	○ 0.202 •	△ 0.213 ↓	△ 0.220 ↓	△ 0.221 ↓	△ 0.226 ↓	△ 0.211 ↓	△ 0.220 ↓	△ 0.222 ↓	△ 0.219 ↓	△ 0.221 ↓	△ 0.224 ↓
0.90	○ 0.201 •	○ 0.204 •	○ 0.208 •	○ 0.207 ↓	△ 0.209 ↓	○ 0.202 •	○ 0.207 •	△ 0.212 ↓	△ 0.213 ↓	○ 0.205 •	○ 0.208 •
Avg. gain over MP		△ 8.2 %	△ 17.60 %	△ 19.07 %	△ **21.23 %**	△ 7.9 %	△ 18.7 %	△ 18.7 %	△ 19.3 %	△ 11.3 %	△ 18.0 %
Highest gain over MP		△ 15.59 %	△ 29.36 %	△ 32.51 %	△ 34.00 %	△ 14.94 %	△ 29.54 %	△ 26.31 %	△ 32.95 %	△ 18.74 %	△ 28.26 %
Avg. gain over FE		–	△ 8.7 %	–	○ 1.8 %	–	△ **10.0** %	–	○ 3.4 %	–	△ 6.0 %
Highest gain over FE		–	△ **21.0** %	–	△ 4.7 %	–	△ 19.2 %	–	△ 7.0 %	–	△ 13.9 %

△ and ○ indicate significant improvement and statistical tie, respectively, with 95 % confidence

↑, ↓ and • indicate that the result is an improvement, decline, or remained the same considering the previous ϵ value

Bold numbers denote the best performance for each sorting criterion across the range of ϵ values (top), as well as the overall best average and highest gains over MP and FE among all sorting criteria (bottom)

Table 2 MRR numbers for the MP algorithm

	Random	Degree		Betweenness		Eigenvector		Clustering coefficient		PageRank	
ϵ	–	FE	KWM	FE	KWM	FE	KWM	FE	KWM	FE	KWM
0.00	0.205 •	0.205 •	0.205 •	0.205 •	0.205 •	0.205 •	0.205 •	0.205 •	0.205 •	0.205 •	0.205 •
0.01	○ 0.207 •	○ 0.206 •	△ 0.218 ↑	△ 0.220 ↑	△ 0.220 ↑	○ 0.209 •	△ 0.214 ↑	△ 0.225 ↑	△ 0.215 ↑	○ 0.208 •	△ 0.219 ↑
0.05	○ 0.206 •	○ 0.207 •	△ 0.232 ↑	△ 0.230 ↑	△ 0.242 ↑	○ 0.209 •	△ 0.242 ↑	△ 0.250 ↑	△ 0.236 ↑	○ 0.209 •	△ 0.240 ↑
0.10	○ 0.207 •	○ 0.212 •	△ 0.249 ↑	△ 0.246 ↑	△ 0.252 ↑	○ 0.212 •	△ 0.252 ↑	△ 0.256 ↑	△ 0.249 ↑	△ 0.218 ↑	△ 0.250 ↑
0.20	○ 0.207 •	△ 0.219 ↑	△ **0.265** ↑	△ 0.264 ↑	△ 0.265 ↑	△ 0.221 ↑	△ 0.259 ↑	△ 0.258 ↑	△ 0.261 ↑	△ 0.228 ↑	△ 0.253 ↑
0.30	○ 0.207 •	△ 0.231 ↑	△ 0.255 ↓	△ **0.273** ↑	△ 0.270 ↑	△ 0.228 ↑	△ **0.265** ↑	△ 0.254 ↓	△ **0.272** ↑	△ 0.243 ↑	△ 0.258 ↑
0.40	○ 0.207 •	△ 0.236 ↑	△ 0.254 ↓	△ 0.260 ↓	△ 0.265 ↓	△ 0.234 ↑	△ 0.260 ↓	△ 0.253 ↓	△ 0.270 ↓	△ 0.238 ↓	△ **0.262** ↑
0.50	○ 0.208 •	△ 0.234 ↓	△ 0.252 ↓	△ 0.258 ↓	△ 0.265 •	△ 0.234 •	△ 0.256 ↓	△ 0.248 ↓	△ 0.260 ↓	0.246 ↑	△ 0.257 ↓
0.60	○ 0.208 •	△ 0.233 ↓	△ 0.246 ↓	△ 0.250 ↓	△ 0.253 ↓	△ 0.232 ↓	△ 0.250 ↓	△ 0.246 ↓	△ 0.246 ↓	△ 0.241 ↓	△ 0.247 ↓
0.70	○ 0.207 •	△ 0.230 ↓	△ 0.235 ↓	△ 0.238 ↓	△ 0.245 ↓	△ 0.225 ↓	△ 0.238 ↓	△ 0.237 ↓	△ 0.235 ↓	△ 0.238 ↓	△ 0.235 ↓
0.80	○ 0.207 •	△ 0.218 ↓	△ 0.225 ↓	△ 0.226 ↓	△ 0.232 ↓	△ 0.216 ↓	△ 0.225 ↓	△ 0.227 ↓	△ 0.224 ↓	△ 0.227 ↓	△ 0.222 ↓
0.90	○ 0.207 •	○ 0.210 •	○ 0.213 •	○ 0.212 •	△ 0.214 ↓	○ 0.208 •	○ 0.213 •	△ 0.217 ↓	△ 0.218 ↓	○ 0.210 •	○ 0.213 •
Avg. gain over MP		△ 8.0 %	△ 17.1 %	△ 18.8 %	△ **20.8 %**	△ 7.7 %	△ 18.7 %	△ 18.4 %	△ 19.1 %	△ 11.1 %	△ 17.8 %
Highest gain over MP		△ 15.26 %	△ 29.07 %	△ 33.12 %	△ 31.82 %	△ 14.43 %	△ 29.36 %	△ 26.02 %	△ 32.79 %	△ 19.78 %	△ 27.79 %
Avg. gain over FE		–	△ 8.5 %	–	○ 1.7 %	–	△ **10.11** %	–	○ 0.5 %	–	△ 6.0 %
Highest gain over FE		–	△ **20.1** %	–	△ 5.3 %	–	△ 19.2 %	–	△ 7.2 %	–	△ 14.8 %

The symbols follow the same meaning as in Table 1

Bold numbers denote the best performance for each sorting criterion across the range of ϵ values (top), as well as the overall best average and highest gains over MP and FE among all sorting criteria (bottom)

Table 3 Kendall's τ pairwise correlation for different centrality metrics

	PageRank	Betweenness	Clustering coefficient	Eigenvector	Degree
PageRank	1.00000	0.00365	−0.00128	0.00017	−0.00053
Betweenness	0.00365	1.00000	−0.00069	0.00042	0.00048
Clustering coefficient	−0.00128	−0.00069	1.00000	−0.00204	−0.00148
Eigenvector	0.00017	0.00042	−0.00204	1.00000	−0.00011
Degree	−0.00053	0.00048	−0.00148	−0.00011	1.00000

baseline recommendation algorithms. Note, however, that simply exploiting more feedback does not necessarily guarantee the best overall performance, as shown by the blue curves. Indeed, we observe that a value of $\epsilon \approx 0.3$ is generally the best trade-off between exploration and exploitation for all the considered instantiations of our approach, in which case the observed gain on top of the three recommendation baselines is maximized. Overall, these results answer question Q2, by further attesting the effectiveness of our explore-then-exploit approach for DDS recommendation. In Sects. 4.2.2 and 4.2.3, we further investigate the reasons behind such an effective performance.

4.2.2 Impact of sorting criteria

The previous section demonstrated the effectiveness of our approach at improving the performance of three representative recommendation algorithms from the literature in a DDS scenario. In this section, we address our question Q3, by investigating the impact of the various sorting criteria proposed in Sect. 3.1 on the overall effectiveness of our approach.

To answer question Q3, Tables 1 and 2 provide a breakdown analysis of our approach in terms of MAP and MRR, respectively. In each table, we show the performance of our approach on top of the MP baseline, which was the best performing recommendation algorithm in Sect. 4.2.1, for different choices of sorting criteria, splitting strategy, and the ϵ parameter. As sorting criteria, besides the aforementioned Betweennes, Degree, Eigenvector, Clustering Coefficient, and PageRank centrality metrics, introduced in Sect. 3.1, we further consider a Random criterion as a baseline. To assess the effectiveness of our proposed KWM splitting strategy, we compare it to the FE strategy proposed by Lacerda et al. (2013). In particular, KWM is deployed with $k = 5$, which delivers a robust performance, as we will show in Sect. 4.2.3. Regarding the ϵ values, recall that $\epsilon = 0$ corresponds to the performance obtained by the MP baseline alone, without any feedback information. For each performance number, we use the symbols $\triangle$ and $\circ$ to indicate significant improvements and statistical ties, respectively, according to a paired t-test with $p < 0.05$. We also use the symbols $\uparrow$, $\downarrow$ and $\bullet$ to indicate that the corresponding result is an improvement, decline, or remained the same considering the previous ϵ value. Finally, at the bottom of each table, we provide summary figures for the average and highest gains of our approach compared to the MP baseline, as well as the average and highest gains of KWM compared to the FE splitting strategy.

In Sect. 4.2.1, we showed that our proposed approach consistently improves on top of the different recommendation baselines for various choices of sorting criteria. To address question Q3, we further investigate the impact of each of these criteria on the performance of our approach. From Tables 1 and 2, we first note that the use of centrality metrics as

sorting criteria is notably effective. Indeed, using the Random baseline as a sorting criterion results in negligible and no significant improvements compared to the MP baseline with no feedback ($\epsilon = 0$). In contrast, all of the proposed sorting criteria based on centrality metrics significantly improve on top of MP for both MAP and MRR (Tables 1 and 2, respectively) for most values of ϵ. In terms of MAP (Table 1), the Betweenness sorting criterion is the most effective for both the FE ($MAP = 0.264$) and KWM ($MAP = 0.267$) splitting strategies. In terms of MRR (Table 2), Betweenness performs the best with FE ($MRR = 0.273$), whereas Clustering Coefficient performs the best with KWM ($MRR = 0.272$). Once again, for both MAP and MRR, the best performance is consistently obtained with $\epsilon = 0.30$. Recalling question Q3, these results show that the sorting criterion has a significant impact on the performance of our explore-then-exploit recommendation approach, with Betweenness giving the overall best performance. In particular, for the range of ϵ values, Betweenness gives a 19.07 % average (32.51 % highest) improvement in terms of MAP on top of MP when using the FE splitting strategy, and 21.23 % (34 % highest) when using the KWM strategy. In terms of MRR, the average gains for Betweenness are 18.8 % (33.12 % highest) and 20.8 % (31.82 % highest) with FE and KWM, respectively. The highest MRR gain over MP across all sorting criteria is observed for Clustering Coefficient with the KWM splitting strategy: 32.79 %. These observations are in line with the results reported by Bellogin et al. (2011), who noted the effectiveness of Betweenness and Degree as weights for combining ensembles of recommender systems in a hybrid approach.

A natural question for further improving the performance of our MAB approach is whether we could combine multiple sorting criteria as arms. To this end, as discussed in Sect. 2, these multiple arms should be independent of one another as much as possible, so as to provide complementary alternatives for exploration. In order to assess the extent to which our considered sorting criteria are independent of one another, we quantify their pairwise correlation using the well-known Kendall's τ metric (Baeza-Yates and Ribeiro-Neto 2008), as reported in Table 3. From the table, we observe that all τ values are very close to 0, showing that the rankings generated by the various considered centrality metrics are indeed not correlated. As previously discussed, this result attests the potential of the several considered centrality metrics as arms within our MAB approach. Harnessing this potential is a direction for future work.

4.2.3 Impact of splitting strategies

The previous section analyzed the impact of the sorting criterion component on the effectiveness of our proposed approach. In this section, we complement this analysis and address our research question Q4, by assessing the impact of the splitting strategy component. In particular, to address question Q4, we contrast the performance of our approach using the KWM splitting strategy, introduced in Sect. 3.2, to its performance using the FE strategy, proposed by Lacerda et al. (2013), across the five considered sorting criteria. From Tables 1 and 2, we observe that the KWM strategy consistently improves compared to FE for almost all sorting criteria and all values of ϵ. The average gains in terms of MAP range from 1.8 to 10.0 % and are significant in most cases (the only exceptions are when using the Betweenness and Clustering Coefficient sorting criteria, in which cases the average gains are not significant). In turn, the highest MAP improvements range from 4.7 % (Betweenness) to 21 % (Degree). In terms of MRR, the average gains range from 0.5 to 10.11 %, which are once again significant in most cases, except for Betweenness and Clustering Coefficient. Similarly to the MAP results, the highest gains in terms of MRR

range from 5.3 % (Betweenness) to 20.1 % (Degree). Recalling question Q4, these results attest the effectiveness of our proposed KWM splitting strategy and its positive impact on the overall performance of our recommendation approach.

To better understand the added benefit of our KWM splitting strategy, we further analyze the co-purchase sub-graph comprising only the customers in the exploration set. In particular, we compare co-purchase sub-graphs generated for different window sizes k. Recall from Sect. 3.2 that the window size k determines the number of customer chunks induced from the ranking of all customers according to a given centrality metric. In particular, this fixed-size window splitting strategy was proposed based on the intuition that, in highly connected sub-graphs, central customers are usually connected to one another. In turn, already connected customers are likely to influence a common set of other customers during exploitation, which is a suboptimal behavior. In contrast, selecting more dispersed customers during exploration has the potential to influence a larger set of customers during exploitation. To quantify this intuition, we hypothesize that the more dispersed the explored customers the less dense the resulting co-purchase sub-graph. To test this hypothesis, we define the density of a sub-graph $G(V, E)$ as the number of edges in G divided by the maximum possible number of edges in G. Since our co-purchase sub-graphs are undirected, the density is computed as:

$$Density(G) = \frac{2 \times |E|}{V \times (V-1)}, \tag{7}$$

where V and E denote the sets of vertices and edges that comprise the graph G, respectively.

Figure 10 shows the density of the co-purchase sub-graphs induced by our KWM splitting strategy for different values of k and various sorting criteria. Note that $k = 1$ equates to our baseline FE splitting strategy, described in Sect. 3.2. In each plot, we also include curves for different values of ϵ, denoting different fractions of customers selected for exploration. Taking the first plot in Fig. 10 (Degree) as a canonical example, we confirm our hypothesis that the larger the window size k, the less dense the exploration sub-graph, and hence the higher the probability that the customers' feedback will influence other customers during exploitation. In particular, a window size $k \approx 5$ provides an already effective dispersion of exploration customers for all considered sorting criteria. Recalling question Q4, this result further demonstrates the potential benefit of our KWM strategy for splitting customers into exploration and exploitation groups.

5 Time performance

The experimental results in Sect. 4 demonstrated the effectiveness of our proposed explore-then-exploit approach for improving the recommendation of deals in a DDS scenario. In this section, we further assess the time performance of our proposed approach, and analyze the four steps of the process presented in Sect. 3. We start by sorting customers off-line according to their purchase history data as illustrated in Fig. 1. This step is divided into three phases: building the co-purchase graph, calculating the centrality metrics, and sorting users according to the metrics. The cost to build the graph is $O(c + n)$, where c is the number of customers and n is the number of deals. The centrality metrics were calculated using the implementations available in the SNAP library.[12] The most expensive metric is Betweenness, which is

[12] http://www.snap.stanford.edu/snap/.

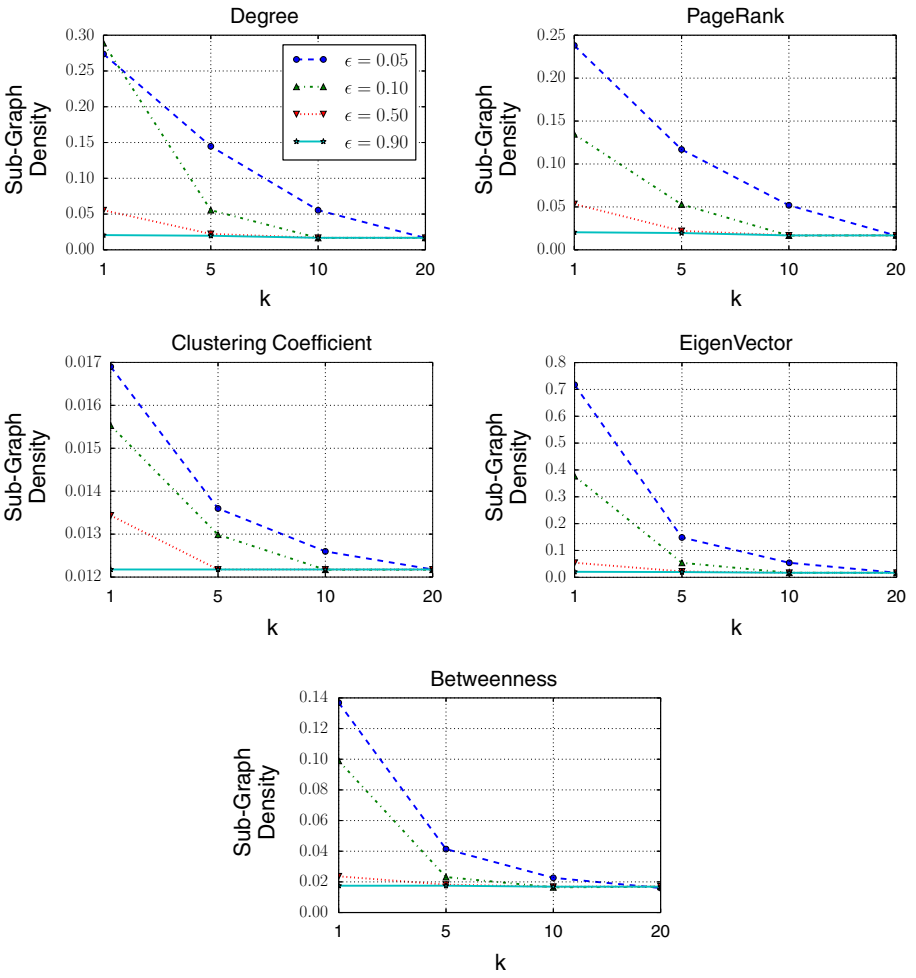


Fig. 10 Co-purchase network density for different sorting criteria and window size k

$O(c^2 \log c + ce)$, where e is the number of edges in the co-purchase network. The phase of sorting customers is $O(c \log c)$. Therefore, the overall complexity of the sorting step is $O(c^2 \log c + ce)$. Recall that this step is performed off-line.

The second step of our approach is the splitting of customers into exploration and exploitation sets, illustrated in Fig. 2. Our FE strategy is $O(1)$ because it does not re-rank the previously computed ranking of customers. The KWM strategy is $O(ck \log k)$ (Knuth 1998), where k is the number of chunks. The splitting step is also performed off-line.

The third step of our approach is the recommendation process for customers in the exploration set, illustrated in Fig. 3. The most expensive recommendation algorithm used in our investigation is WRMF, which is $O(c + N)$, where N is the number of non-zero observations in the customer/deal matrix (Hu et al. 2008). The online part of this step refers to sending emails for each customer and updating the co-purchase network. The running

time to send emails is $O(c)$. The running time to update the co-purchase network is $O(p)$, where p is the number of co-purchases that happened during exploration.

The last step of our process is the recommendation process for customers in the exploitation set, illustrated in Fig. 4. We used the recommender trained during exploration to compute recommendations. Hence, the additional step here is the re-ranking of recommendations using the co-purchase network. This equates to a sorting procedure that is $O(r \log r)$, where r is the number of deals recommended to customers. The overall running time of the off-line and online methods are hence $O(c^2 \log c + ce)$ and $O(c)$, respectively.

6 Conclusions and future work

This paper focused on the problem of suggesting relevant and appealing products and services to potential customers, considering the particularly challenging scenario of daily deals recommendation. Specifically, we considered the task of sending personalized email messages featuring potentially relevant deals to customers. In this case, we can impose a restriction on the order according to which customers receive their messages, that is, some customers receive their messages before others. Based upon this characteristic of the problem, we proposed an explore-then-exploit recommendation approach devised to: (1) gather feedback from customers that receive their messages first (i.e., during the exploration phase), and (2) use the gathered feedback in order to send personalized messages to the remaining customers (i.e., during the exploitation phase). An important advantage of our approach is that it is agnostic to the underlying recommendation algorithm used to send personalized messages, i.e., we are able to use any recommendation algorithm proposed in the literature.

There is a trade-off between exploration and exploitation, in the sense that the more customers are explored, more feedback is gathered but less feedback is actually exploited. To deal with this trade-off, we proposed to sort customers according to their centrality in an evolving co-purchase network, so as to explore the feedback of customers with a higher influence on the remaining customers. To achieve a proper balance between the amount of feedback acquired during exploration and the amount of feedback indeed used during exploitation, we proposed a splitting strategy aimed at selecting a dispersed set of exploration customers across the co-purchase centrality spectrum. Using a sampled dataset covering a period of 2 months in late 2012 from Peixe Urbano, the largest DDS in Brazil, we showed that: (1) the recommendation performance of traditional recommendation algorithms in a DDS scenario is rather limited, primarily because of the ephemeral nature of the available deals; (2) our proposed explore-then-exploit approach is very effective in this scenario, providing improvements ranging from 14 to 34 % in mean average precision and MRR compared to existing algorithms.

Our proposed directions for future investigations include: (1) alternative MAB algorithms (Li et al. 2010a, 2011), devised to deal with the exploration-exploitation trade-off, (2) strategies to remove obsolete edges from the co-purchase network, so that more recent interactions are taken into account, and (3) large-scale and online controlled experiments with A/B testing (a different bandit approach) (Kohavi et al. 2012; Kohavi 2012; Rodriguez et al. 2012), enabling us to evaluate our approach using live customer interactions.

Acknowledgments This work is partially supported by the National Institute of Science and Technology for the Web, MCT/CNPq Grant 57.3871/2008-6, and by the authors' individual grants and scholarships from CAPES and CNPq. We thank Peixe Urbano for providing the data used in the experiments.

References

Aizenberg, N., Koren, Y., & Somekh, O. (2012). Build your own music recommender by modeling internet radio streams. In *Proceedings of the 21st international conference on world wide web*, (pp. 1–10).

Arabshahi, A. (2011). Undressing Groupon: an analysis of the Groupon business model. http://www.ahmadalia.com/blog/2011/01/undressing-groupon.html

Auer, P., Cesa-Bianchi, N., & Fischer, P. (2002). Finite-time analysis of the multiarmed bandit problem. *Machine Learning*, *47*(2–3), 235–256.

Azoulay-Schwartz, R., Kraus, S., & Wilkenfeld, J. (2004). Exploitation vs. exploration: Choosing a supplier in an environment of incomplete information. *Decision Support Systems*, *38*(1), 1–18.

Baeza-Yates, R., & Ribeiro-Neto, B. (2008). *Modern information retrieval* (2nd ed.). Addison-Wesley Publishing Company.

Bellogin, A., Castells, P., & Cantador, I. (2011). Self-adjusting hybrid recommenders based on social network analysis. In *Proceedings of the 34th international ACM SIGIR conference on research and development in information retrieval, ACM.* (pp. 1147–1148).

Bifet, A., Holmes, G., Kirkby, R., & Pfahringer, B. (2010). MOA: Massive online analysis. *Journal of Machine Learning Research*, *11*(1), 1601–1604.

Boccaletti, S., Latora, V., Moreno, Y., Chavez, M., & Hwang, D.-U. (2006). Complex networks: Structure and dynamics. *Physics Reports*, *424*(4), 175–308.

Bonacich, P. (1987). Power and centrality: A family of measures. *The American Journal of Sociology*, *92*(5), 1170–1182.

Borgatti, S. (2005). A graph-theoretic perspective on centrality. *Social Networks*, *28*(4), 466–484.

Bottou, L., Peters, J., Candela, J., Charles, D. X., Chickering, M., Portugaly, E., et al. (2013). Counterfactual reasoning and learning systems: The example of computational advertising. *Journal of Machine Learning Research*, *14*(1), 3207–3260.

Bouneffouf, D., Bouzeghoub, A., & Gançarski, A. L. (2012). A contextual-bandit algorithm for mobile context-aware recommender system. In *Proceedings of the 19th international conference on neural information processing systems*, (pp. 324–331).

Byers, J., Mitzenmacher, M., & Zervas, G. (2012a). Daily deals: Prediction, social diffusion, and reputational ramifications. In *Proceedings of the 5th ACM international conference on web search and data mining*, (pp. 543–552).

Byers, J. W., Mitzenmacher, M., Potamias, M., & Zervas, G. (2011). A month in the life of groupon. arXiv preprint arXiv:1105.0903.

Byers, J. W., Mitzenmacher, M., & Zervas, G. (2012b). The Groupon effect on Yelp ratings: A root cause analysis. In *Proceedings of the 13th ACM conference on electronic commerce*, (pp. 248–265).

Chakrabarti, D., Kumar, R., Radlinski, F., & Upfal, E. (2008). Mortal multi-armed bandits. In *Proceedings of the 22nd annual conference on neural information processing systems*, (pp. 273–280).

Conry, D., Koren, Y., & Ramakrishnan, N. (2009). Recommender systems for the conference paper assignment problem. In *Proceedings of the 3rd ACM conference on recommender systems*, (pp. 357–360).

Cremonesi, P., & Koren, Y. (2010). Performance of recommender algorithms on top-n recommendation tasks. ... *conference on recommender* ....

Dholakia, U. M. (2010). How effective are Groupon promotions for business. http://www.ruf.rice.edu/~dholakia

Edelman, B., Jaffe, S., & Kominers, S. (2011). To Groupon or not to Groupon: The profitability of deep discounts. *Harvard Business School NOM unit working paper*, (pp. 11–063).

Even-Dar, E., Mannor, S., & Mansour, Y. (2006). Action elimination and stopping conditions for the multi-armed bandit and reinforcement learning problems. *The Journal of Machine Learning Research*, *7*(1), 1079–1105.

Feng, J., Bhargava, H., & Pennock, D. (2007). Implementing paid placement in web search engines: Computational evaluation of alternative mechanisms. *INFORMS Journal on Computing*, *19*(1), 37–49.

Freeman, L. (1979). Centrality in social networks: Conceptual clarification. *Social Networks*, *1*(3), 215–239.

Guha, S., & Munagala, K. (2007). Multi-armed bandits with limited exploration. In *Proceedings of the annual symposium on theory of computing*, (pp. 1–19).

He, D., Chen, W., Wang, L., & Liu, T.-Y. (2013). Online learning for auction mechanism in bandit setting. *Decision Support Systems*, *56*, 379–386.

Hu, Y., Koren, Y., & Volinsky, C. (2008). Collaborative filtering for implicit feedback datasets. In *Proceedings of the 8th IEEE international conference on data mining*, (pp. 263–272).

Kaelbling, L. P., Littman, M. L., & Moore, A. W. (1996). Reinforcement learning: A survey. *Journal of Artificial Intelligence Research*, *4*(1), 237–285.

Kauffman, R. J., & Wang, B. (2001). New buyers' arrival under dynamic pricing market microstructure: The case of group-buying discounts on the internet. *Journal of Management Information Systems*, *18*(2), 157–188.

Knuth, D. E. (1998). Sorting and searching. *The art of computer programming* (2nd ed., Vol. 3). Redwood City, CA: Addison Wesley Longman Publishing Co., Inc.

Koenigstein, N., Dror, G., & Koren, Y. (2011). Yahoo! music recommendations: Modeling music ratings with temporal dynamics and item taxonomy. In *Proceedings of the 5th ACM conference on recommender systems*, (pp. 165–172).

Kohavi, R. (2012). Online controlled experiments: Introduction, learnings, and humbling statistics. In *Proceedings of the 6th ACM conference on recommender systems*, (pp. 1–2).

Kohavi, R., Deng, A., Frasca, B., Longbotham, R., Walker, T., & Xu, Y. (2012). Trustworthy online controlled experiments: Five puzzling outcomes explained. In *Proceedings of the 18th ACM international conference on knowledge discovery and data mining*, (pp. 786–794).

Koren, Y. (2010). Factor in the neighbors: Scalable and accurate collaborative filtering. *ACM Transactions on Knowledge Discovery from Data*, *4*(1), 1–24.

Kumar, V., & Rajan, B. (2012). Social coupons as a marketing strategy: A multifaceted perspective. *Journal of the Academy of Marketing Science*, *40*(1), 120–136.

Lacerda, A., Veloso, A., & Ziviani, N. (2013). Exploratory and interactive daily deals recommendation. In *Proceedings of the 7th ACM conference on recommender systems*, (pp. 439–442).

Lai, T.-L., & Yakowitz, S. (1995). Machine learning and nonparametric bandit theory. *Automatic Control, IEEE Transactions on*, *40*(7), 1199–1209.

Landherr, A., Friedl, B., & Heidemann, J. (2010). A critical review of centrality measures in social networks. *Business and Information Systems Engineering*, *2*(6), 371–385.

Lappas, T., & Terzi, E. (2012). Daily-deal selection for revenue maximization. In *Proceedings of the 21st ACM international conference on information and knowledge management*, (pp. 565–574).

Leskovec, J., & Faloutsos, C. (2006). Sampling from large graphs. In *Proceedings of the 12th ACM SIGKDD international conference on knowledge discovery and data mining*, (pp. 631–636).

Li, L., Chu, W., Langford, J., & Schapire, R. E. (2010a). A contextual-bandit approach to personalized news article recommendation. In *Proceedings of the 19th international conference on world wide web*, (pp. 661–670).

Li, L., Chu, W., Langford, J., & Wang, X. (2011). Unbiased offline evaluation of contextual-bandit-based news article recommendation algorithms. In *Proceedings of the 4th ACM international conference on web search and data mining*, (pp. 297–306).

Li, W., Wang, X., Zhang, R., Cui, Y., Mao, J., & Jin, R. (2010b). Exploitation and exploration in a performance based contextual advertising system. In *Proceedings of the 16th ACM international conference on knowledge discovery and data mining*, (pp. 27–36).

Liu, Y., Li, H., & Hu, F. (2013). Website attributes in urging online impulse purchase: An empirical investigation on consumer perceptions. *Decision Support Systems*, *55*(3), 829–837.

Mahajan, D. K., Rastogi, R., Tiwari, C., & Mitra, A. (2012). Logucb: An explore-exploit algorithm for comments recommendation. In *Proceedings of the 21st ACM international conference on information and knowledge management*, (pp. 6–15).

Maiya, A. S., & Berger-Wolf, T. Y. (2011). Benefits of bias: Towards better characterization of network sampling. In *Proceedings of the 17th ACM SIGKDD international conference on knowledge discovery and data mining*, pp. (105–113).

Mannor, S., & Tsitsiklis, J. N. (2004). The sample complexity of exploration in the multi-armed bandit problem. *The Journal of Machine Learning Research*, *5*, 623–648.

Menezes, G., Almeida, J., Belém, F., Gonçalves, M., Lacerda, A., Moura, E., Pappa, G., Veloso, A., & Ziviani, N. (2010). Demand-driven tag recommendation. In *Proceedings of the 2010 European conference on machine learning and knowledge discovery in databases*, (pp. 402–417).

Miller, B. N., Albert, I., Lam, S. K., Konstan, J. A., & Riedl, J. (2003). Movielens unplugged: Experiences with an occasionally connected recommender system. In *Proceedings of the 8th international conference on intelligent user interfaces*, (pp. 263–266).

Page, L., Brin, S., Motwani, R., & Winograd, T. (1999). The pagerank citation ranking: Bringing order to the web. Technical report 1999–66, Stanford InfoLab.

Pan, R., Zhou, Y., Cao, B., Liu, N. N., Lukose, R., Scholz, M., & Yang, Q. (2008). One-class collaborative filtering. In *Proceedings of the 8th international conference on data mining*, (pp. 502–511).

Radlinski, F., Kleinberg, R., & Joachims, T. (2008). Learning diverse rankings with multi-armed bandits. In *Proceedings of the 25th international conference on machine learning*, (pp. 784–791).

Rendle, S., Freudenthaler, C., Gantner, Z., & Schmidt-Thieme, L. (2009). BPR: Bayesian personalized ranking from implicit feedback. In *Proceedings of the 25th conference on uncertainty in artificial intelligence*, (pp. 452–461).

Ricci, F., Rokach, L., Shapira, B., & Kantor, P. B. (2010). *Recommender systems handbook* (1st ed.). New York, NY: Springer-Verlag New York, Inc.

Robbins, H. (1952). Some aspects of the sequential design of experiments. *Bulletin of the American Mathematical Society*, *58*(5), 527–535.

Rodriguez, M., Posse, C., & Zhang, E. (2012). Multiple objective optimization in recommender systems. In *Proceedings of the 6th ACM conference on recommender systems* (pp. 11–18).

Schein, A., Popescul, A., & Ungar, L. (2002). Methods and metrics for cold-start recommendations. In *Proceedings of the 25th annual international ACM SIGIR conference on research and development in information retrieval*.

Shi, Y., Karatzoglou, A., Baltrunas, L., Larson, M., Oliver, N., & Hanjalic, A. (2012). CLiMF: Learning to maximize reciprocal rank with collaborative less-is-more filtering. In *Proceedings of the 6th ACM conference on recommender systems* (pp. 139–146).

Sutton, R. S., & Barto, A. G. (1998). *Reinforcement learning: An introduction* (Vol. 1). London: Cambridge University Press.

Tran-Thanh, L., Chapman, A., Munoz De Cote Flores Luna, J. E., Rogers, A., & Jennings, N. R. (2010). Epsilon-first policies for budget-limited multi-armed bandits. In *Proceedings of the 24th AAAI conference on artificial intelligence*, (pp. 1211–1219).

Watkins, C. J. C. H. (1989). *Learning from delayed rewards*. PhD thesis.

Ye, M., Sandholm, T., Wang, C., Aperjis, C., & Huberman, B. A. (2012). Collective attention and the dynamics of group deals. In *Proceedings of the 21st international conference companion on world wide web*, (pp. 1205–1212).